A context-aware smart-call-center solution: Improving customer service for online games

L. Luo
J. Liu
L. Shao
W. Lu
M. Ye

Call centers play an important role in the online game industry, using automatic call navigation, information integration, and human agent communication to provide interactive services to assist customers. The unique nature of the online game experience has led to new requirements for the call center communication model. In this paper, we propose a context-aware smart-call-center solution which increases customer satisfaction and provides cost savings to online game operators. By leveraging VoIP (voice over Internet Protocol) technology and transmitting contextual information for the player and the game, the smart call center enables a context-aware service capability, helping call-center agents to provide efficient customer support and reducing overall service time and the number of call-center agents. In addition to games, the smart-call-center implementation described in this paper can be used in various solutions that provide instant assistance to customers of online businesses.

INTRODUCTION

A call center (also known as a contact center) is an interactive value-added service system providing assistance and consulting to customers through interactive call navigation, CTI (Computer Telephony Integration) technology, and human agent communication. Customers can access a call center using a phone, FAX machine, wireless device, or computer. Call centers are extensively deployed in many industries, including banking, product support, and telemarketing.[1–3] Call-center services are used for inbound and outbound calls (i.e., those from and to customers), with the former being more common.

In the online game industry, call centers have increased in importance. In an iResearch 2004 survey,[4] 11.5 percent of game players who abandoned a game did so because of poor customer service. Shanda, the largest online game company in China, owes its exponential business growth to the early deployment of an advanced call center.[5,6] High-quality customer service can significantly enhance the attractiveness of the game experience. In the growing online game market, providing unique services is a critical way for companies to acquire and increase market share and differentiate

©Copyright 2006 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the Journal reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to republish any other portion of the paper must be obtained from the Editor. 0018-8670/06/$5.00 © 2006 IBM

themselves. As elsewhere, online games represent a growing market in China.[7]

In an online-game call center, the agents (i.e., customer service representatives or CSRs) are sometimes game experts as well. Their daily work differs from that of CSRs in traditional businesses. In traditional contexts such as banking and product support, customer requirements are relatively simple (e.g., obtaining account information, product troubleshooting, and learning about products). In the case of online games, many requests focus on instant in-game support, requiring game experts to provide timely service based on the runtime game status.

Current online-game call centers provide primitive functions such as account-information retrieval, game coin trading, and answering general product questions. Many players consider these functions to be insufficient. For instance, new players may not be familiar with the rules and current maps of the game, and this information is not easily communicated over the phone or in text. In addition, some assistance must be given in close temporal coordination with game play, requiring contextual information, but some of this information, including user ID, game name, and other variables is hard to communicate in a timely manner using traditional means. "Context" here means the information that describes the situation of the game client in the specific game environment. Game players need efficient communication when they connect to a call center.

In this paper, we propose a smart-call-center (SCC) solution for online-game call-center operators that increases productivity and reduces the total cost of ownership (TCO). By leveraging the VoIP (voice-over-IP [Internet Protocol]) infrastructure and a game context enabler, the SCC enables context-aware service-enabling call-center agents to provide more efficient customer support. The multiple functions of the SCC are integrated into a consistent view in a single window, facilitating the use of diverse information by the call-center agent. The prototype of the SCC is described in detail, and the results of quantitative analysis are presented to demonstrate the SCC's ability to reduce overall service time and the number of call center agents. The context-aware SCC principles presented here can be applied to upgrading existing call centers or building new call centers for many online businesses.

Call-center background

Since their emergence several decades ago, call-center services have gone through two revolutionary transitions. The first transition was from the primitive "hotline" dial-in model to the automatic reaction model based upon CTI (Computer Telephony Integration). In addition, it included voice transmission within call centers based upon a TDM (Time Division Modulation) circuit-switched network and access through the public switched telephone network (PSTN). The second transition is ongoing, as the current traditional TDM-based model is migrating to IP (shown in ***Figure 1***) and as traditional voice communication is being extended to include multichannel contact service with technologies such as VoIP, e-mail, and instant messaging (IM). IP-based call centers can be further categorized into two network types: intra-IP and omni-IP, which differ in whether user access occurs through traditional PSTN-based phone service or through IP phone service. Deployments sometimes use a transitional combination of IP and TDM.

TDM-based and IP-based call centers have many similar basic functional components:

- ACD (automatic call distributor)—This component conducts inbound call queuing and dispatches calls to the appropriate agents. In a traditional call center, this is a hardware device; in an IP-based one, it is usually a software module.
- PBX (private branch exchange)—After inbound calls are dispatched, the hardware PBX routes the TDM calls to agents. For IP packet transmission, no PBX is required.
- IVR (interactive voice response)—This component provides automatic voice interactive responses to customers and can direct them to different services.
- CTI—This component integrates computer interaction with the telephone exchange. For example, at the same time that an agent accepts a call, a window displaying call-related data may pop up on the agent's computer screen.
- CRM (customer relationship management)—This component stores and manages customer-related information for analysis and to provide enhanced and personalized service.

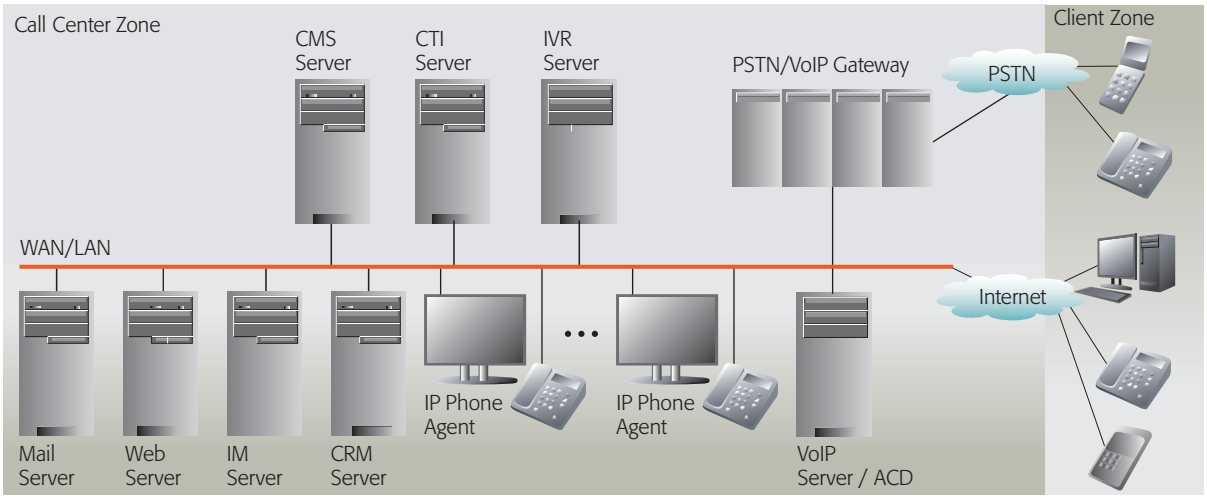

Figure 1
IP call center architecture

Generally, a customer call first goes through the call gateway. Then an ACD functional module places the call in the queue and evaluates the status (busy or available) of the agents. If there is an available agent, the call is distributed to the agent through PBX or IP transmission. In an IP call center, TDM signals from the PSTN network are translated to IP data. Frequently, an IVR system may respond to the user before the call is established between the user and a real agent. The agent's "desktop" can be a traditional TDM phone or an IP one as in Figure 1. Currently, in both cases, voice signals and Web data are not efficiently integrated, which may result in inefficiency in some scenarios. This limitation is discussed in the next section.

A call center service's design and performance can be evaluated using quantitative and qualitative measurements.[8,9] A well-accepted quantitative measurement set includes the following variables:

- *Average answer speed*—The time interval beginning when a user initiates a call and ending when an agent answers the phone.
- *Abandonment rate*—Abandonment occurs when a waiting user decides to hang up the phone before an agent has answered.
- *Service level*—The rate at which an agent can successfully serve a user.
- *Average service time*—The average time for each agent to serve a user.
- *Number of lines*—The total number of lines or trunks in a call center. This is based on the physical implementation of the call center.
- *Number of agents*—The concurrent number of agents during a given period of time. This number is dynamic, depending on the user call arrival rate, and is less than the number of lines.

The qualitative measurements are generally presented in the manner of survey statistics or in a comment list. These comments are mainly based on the subjective feelings of customers, such as whether the agent understands the customer's meaning quickly and precisely, is professional, can give specific suggestions and solve problems successfully, and so on.

Requirements for online-game call centers

IP call centers have many benefits, such as remote access by agents and convenience in upgrading and expansion. Nonetheless, although some call-center providers advocate IP-based call centers, many customer companies are reluctant to adopt them as their call center solution. This is because many existing call centers were deployed long before the advent of IP call centers, and the cost of a migration from legacy systems (e.g., hardware-based PBX/ACD systems, TDM networks, etc.) is high. Also, because the major access method in many industries, such as banking, product service, and government service,[1,2] is still based on traditional PSTN

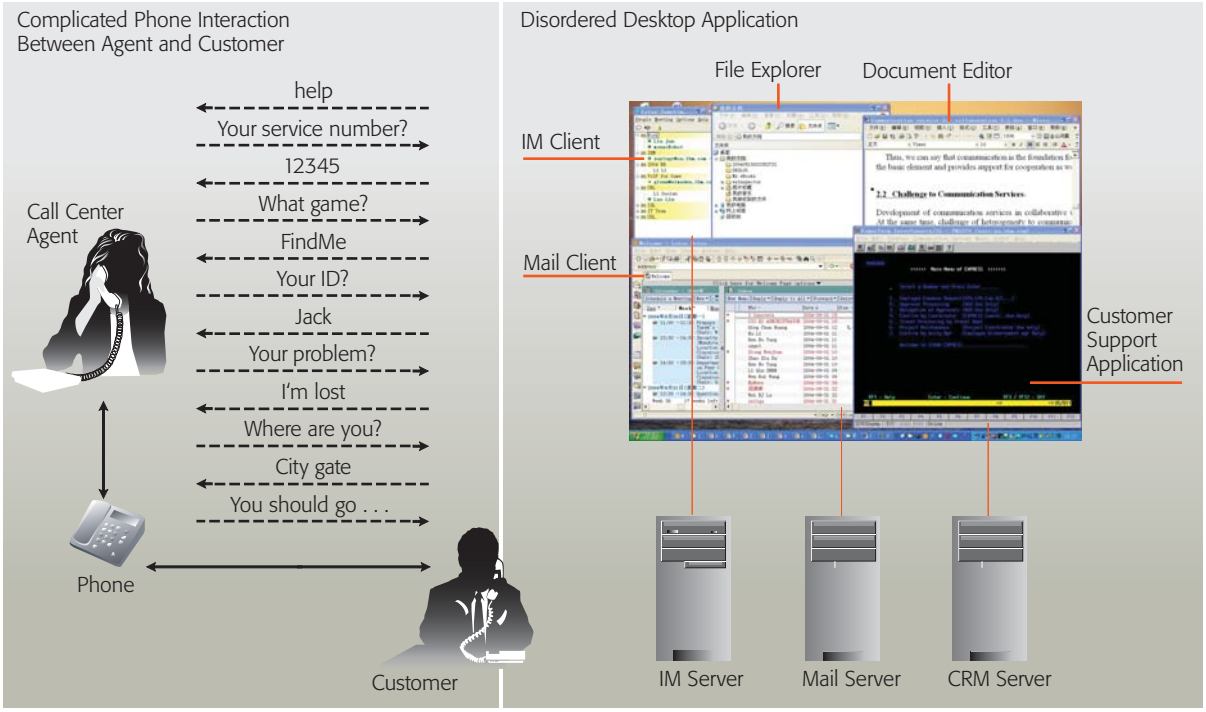

Figure 2
Existing solution

access instead of Web access, a TDM/CTI-based infrastructure is considered sufficient for their business. Furthermore, many companies don't have explicit requirements concerning the combination of voice communication and Web service data.

In contrast with many industries using traditional call centers, the online game industry is young and possesses some unique characteristics. Firstly, because game access is already Web-based, IP access to a call center service is very natural. Secondly, a significant portion of game players solicit help for instant in-game issues such as upgrading hints, guidance from "top guns," map information, and so forth. As a result, a "just-in-time" online help system is critical. Thirdly, because many start-up game operators have just entered or plan to enter this emerging market without any legacy call center deployment, they are willing to adopt an advanced call center solution. These characteristics make the adoption of IP-based call centers in the game industry more natural, and the convergence of voice and game-related data for these call centers is an essential requirement. This is the motivation of the proposed context-aware SCC.

A typical scenario for an online-game call-center service is shown in ***Figure 2***. Jack, an online game player, calls the service because he has gotten lost in the game. With traditional call center service, Jack has to verbally inform the agent about his user ID, game type, service number, approximate location, and so on. The agent can then choose to join the game to help him. This verbal communication of the game parameters is very time-consuming and may cause Jack to abandon the call before he gets help. It also results in lengthy average service times, so that the service level and customer satisfaction are negatively affected. In our SCC solution, these problems can be mitigated by transmitting game context and voice data simultaneously so that a great deal of relevant information can be automatically transmitted without verbal description, as shown in ***Figure 3***.

General IP call centers for Web applications have some elements in common with the SCC because the former also combine voice signals with the context of the Web content. Game applications can be considered as one kind of Web application with some unique requirements, and some of the contextual information of the SCC pertains to

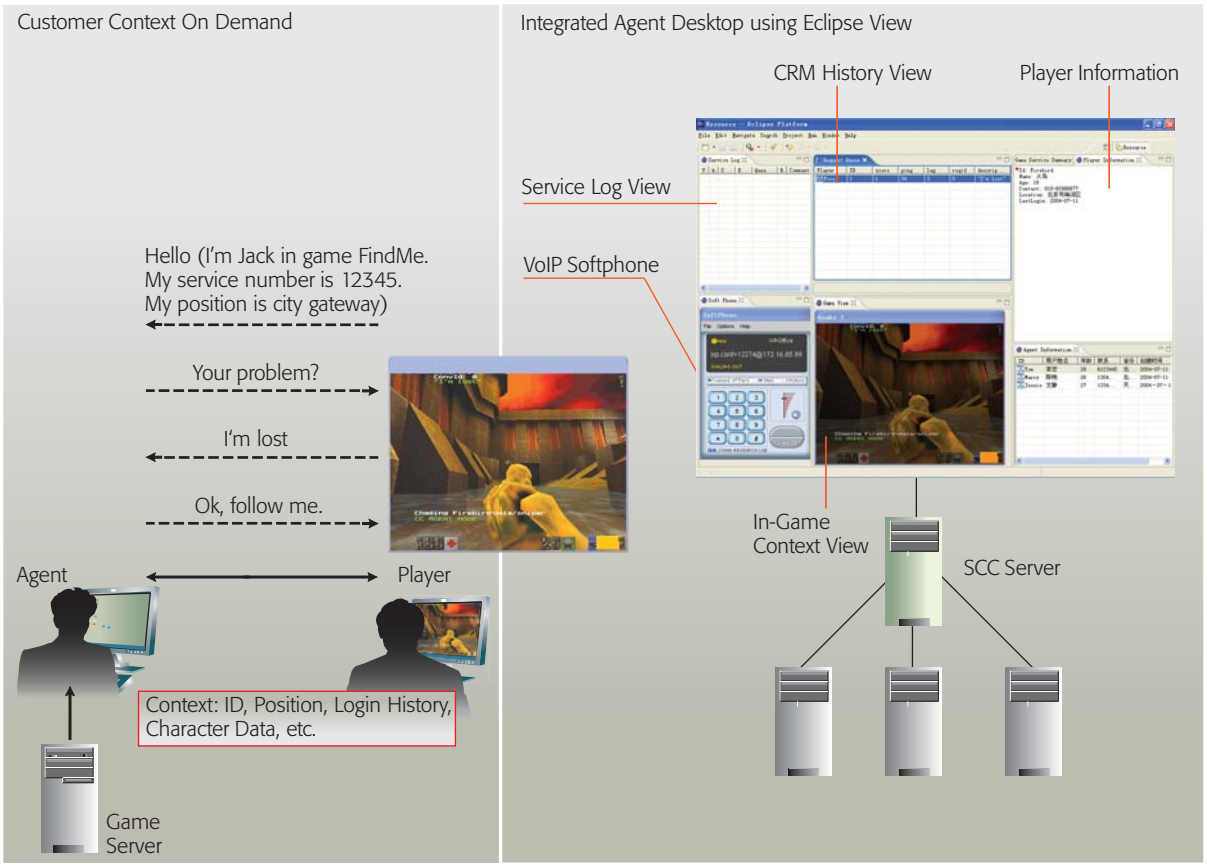

Figure 3
Smart Call Center solution

generic Web applications as well. The major difference between Web application call centers and the SCC exists in the context types and interaction scenarios. In Web applications such as online purchasing, because the agent doesn't need to view a screen environment which is similar to that of the customer, contextual information is mainly used for reference. In the game environment, some contextual information, such as positional coordinates, is used to reconstruct an environment on the agent's screen, thus requiring more complex capabilities for the game agent's equipment in contrast to that of the generic IP agent. Furthermore, the requirement of instant response speed is more important for the online-game call center, requiring a more effective programming model for the call-center server module, agent module, and game server module.

SCC: SYSTEM DESIGN AND IMPLEMENTATION

In this section, we introduce the context-aware SCC solution in three levels: (1) the overall architecture and its differences from the traditional call center, (2) the functional modules in each component system, and (3) the configuration of the solution implementation.

System architecture

The proposed context-aware SCC is an inbound call center that uses VoIP based upon SIP (Session Initiation Protocol[10]) for both customer access and call center agent communication. ***Figure 4*** shows the overall architecture of the SCC and its relationship with the online game operation environment. As shown in the client zone, players can access the SCC in one of two ways, based on their preference, by the traditional PSTN phone or the context-aware IP phone. If they access the SCC through the PSTN, traditional voice communication can be offered to solve some offline questions such as account retrieval or general game introduction, but this is not efficient for many online game questions. The

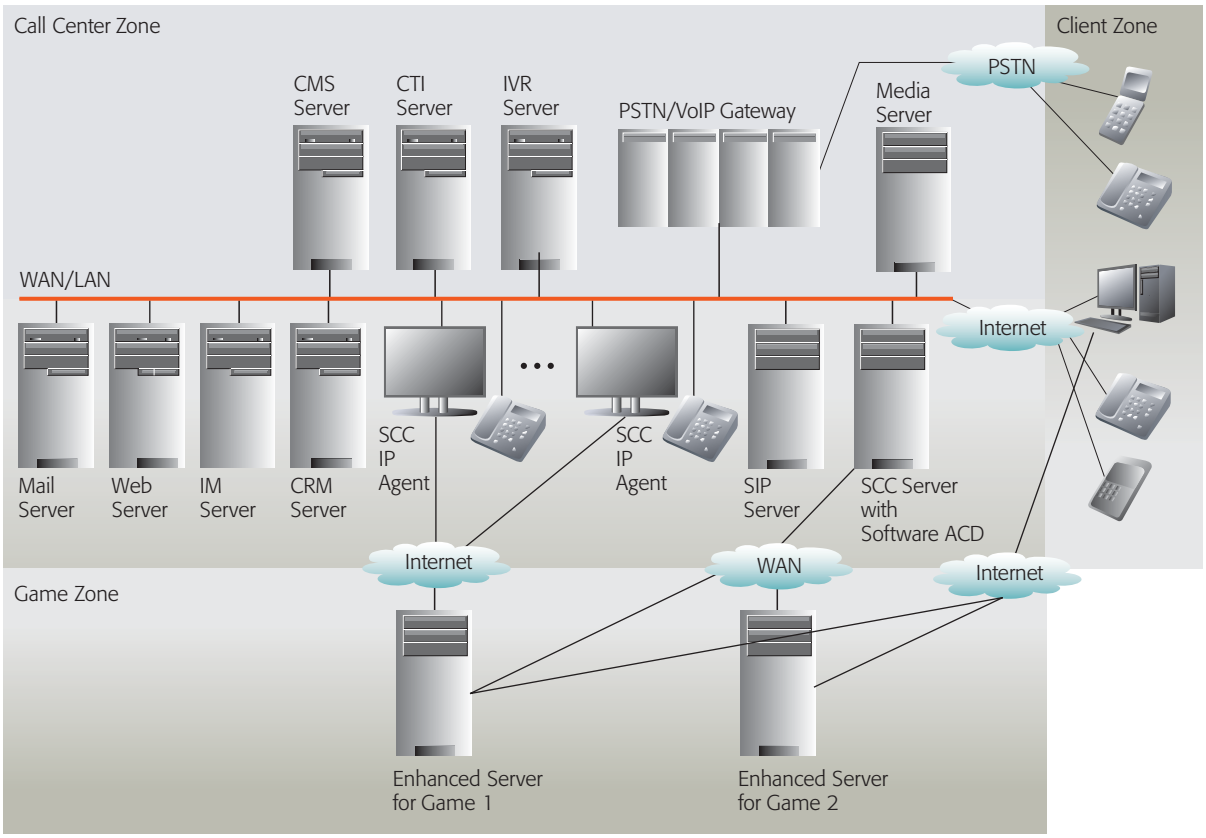

Figure 4
SCC architecture

main differentiations of the SCC solution pertain to the use of IP phone technology.

Aside from information conveyed by requests such as "I want to go to the Enchanted Forest . . ." or "Please give me some hints to upgrade my equipment . . . ," three other types of contextual information that needs to be delivered synchronously can be categorized for online-game scenarios:

Type 1—Offline game-related attributes, such as game name, in-game character ID, character role, digital card information, and so forth. This information could be retrieved from the game client and sent out when a player accesses the call center.

Type 2—Runtime status, such as current location, route, grade, equipment, money, trade history, and so forth. This information can be retrieved either from game clients or game servers, depending on the game design.

Type 3—Players' historical request records, which are stored in a CRM server.

This kind of contextual information can be automatically retrieved from the game client, the runtime game server, or the CRM database in a call center. This is the central idea of our context-aware SCC. With this information, a call center agent can understand the player quickly and give timely assistance.

In the call center outsourcing model, the game operator and the call center operator may belong to different companies, and one call center may provide service to more than one game operator company. As shown in Figure 4, the game server is an enhanced server, which can communicate with the SCC server and share context with it. Besides the components of the IP-based call center shown in Figure 1, some new components are added in the SCC solution: the enhanced game server, enhanced

game client, SCC server, and SCC agent. The SIP server and the media server are also required to provide the VoIP environment.

The enhanced game server manages all game logic operation. Compared with a traditional game server, the enhanced game server also receives and verifies the SCC server's queries for player context information, extracts them, and packages them into context messages that are sent to the SSC server.

The enhanced game client is both a game-rendering client connected to the game server and an embedded VoIP "softphone" (software telephony application) to contact the SCC server. In traditional game clients, a player in a fighting game who wants to initiate an audio conversation has to release a hand to use a physical telephone or standalone VoIP application. This can be difficult or inconvenient for players who are in the midst of fighting a battle. In addition, these stand-alone tools cannot deliver game-related context to the call center. Thus, it is important to embed a voice communication softphone into the game client.

The SCC server is an edge server that receives context-enabled calls from a game client and acquires the corresponding contextual information from the game server. It also takes charge of the software implementation of ACD functions to dispatch the synchronized voice-context calls to an SCC agent. The SCC server is the key component of the context-aware SCC for online games.

The SCC agent has all the functions of a traditional call center agent. In addition, it integrates all the needed functions in a workbench window to assist the agent in serving the game client's request. These functions include a VoIP softphone, CRM database management tools, agent list, service log, and so forth. The agent can automatically get context from the game server or CRM server to investigate the game client's problem. The context-rendering window can be independent of the game window, so that the agent does not have to enter the game to understand the context and provide help. Usually, a game client is also installed, to enable agents to enter the game if needed. If the game server supports it, the representative can act in a "super player" role to follow or guide the requesting player or provide special help.

The SIP server is the essential network element that enables the SCC agent to register location and exchange messages; it uses routing and security policies to deal with various operation environments safely. A typical SIP server supports the functions of user registration, contact redirection, and message routing.

The media server is a network element that supports the processing of audio streams for voice services. This processing includes functions such as merging multiple streams, audio recording and playback, and playing announcements. These functions provide players with services such as interactive announcements and collaborative support.

A sample workflow is presented in ***Figure 5*** to show how the SCC agent interacts with a game player in an instant consulting service. The figure also shows when and where the three types of context described previously are acquired. When a player clicks a button labeled Access Smart Call Center in a game, the call request with client context is sent to the SCC server. The request is queued in the SCC server and is subsequently dispatched to an SCC agent with the context retrieved from the game server and CRM server. The SIP-based conference is then established between the player and the agent through the SIP server. At the same time, the combination of the three types of game contextual information is rendered on the agent's workbench window. The contextual information, game client view, and SIP softphone are arranged as an integrated view in the workbench window, as shown in ***Figure 6***.

Functional modules

Figure 7 shows a component view of the SCC server and client and the enhanced game server and client. Some key components are described next.

The *context definition* component is a block of data describing the format of the game client contextual information. Each game can have its own context definition. In the context-aware SCC for online games, call center agents use the context definition to parse the corresponding context they receive. Context definitions may be described by use of a formal language. In this paper, we use XML Schema to describe the context definition and XML for the contextual information.

The SCC server consists of four modules. The *software ACD module* performs the same logic as a

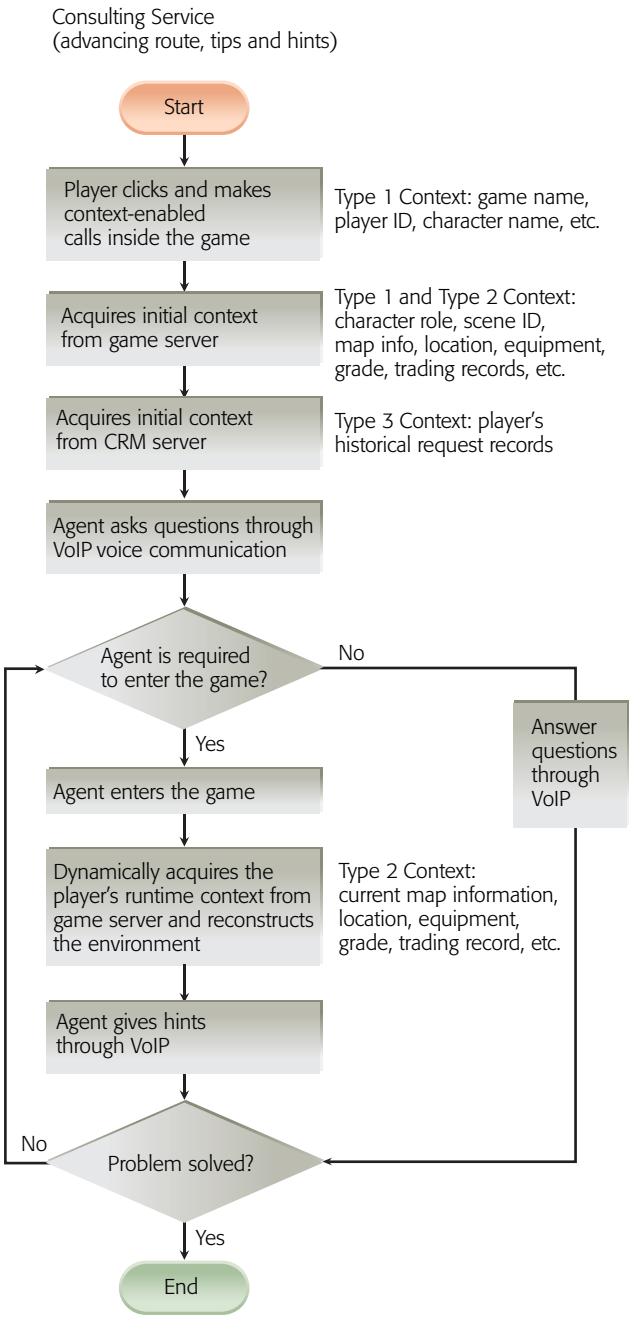

Figure 5
Sample SCC workflow

physical ACD, such as queuing the game clients' requests and selecting an appropriate call center agent to serve the game client's request. The *communication module* accepts calls and context from the IP network. The *context definition retriever* queries and acquires the context definition from the game server and transmits it to a corresponding SCC agent. The *context parser* identifies the collected context and adapts it for the appropriate call center agent, because a call center will most likely run services for various kinds of online games simultaneously.

The SCC agent manages the game client's requests in an integrated environment and consists of six modules. Aside from the basic network interface and the I/O and display modules, the agent consists of the *embedded softphone*, responsible for both VoIP communication and context reception, the *context parser and presenter*, which analyzes received context messages according to the prestored context definition and then renders them in the integrated agent workbench, and the *CRM engine*, which stores related records of a game player into the database.

The enhanced game server is a regular online game server with one added module, a *context extractor*, which extracts runtime contexts and packages them in context messages by using the context definition. The context message may be XML-based. Like the SCC agent, the enhanced game client also includes an embedded SIP softphone to provide convenient in-game access to the SCC server. The softphone module handles both voice signals and context information. The *context packager* in the enhanced game client acquires client-side context and packages it for the softphone to deliver.

The two primary processes of a call center service are (1) the initial context definition downloading (and updating) and (2) the context acquisition and communication establishment for both VoIP and the game environment after a game player clicks on the Access Smart Call Center button.

Context definition

Before an SCC agent can understand the context messages of players, the context definition must be downloaded. For each kind of online game that the call center supports, this process should be conducted at least once during the initiation process and may be repeated to update the context definitions.

The context definition is downloaded from the enhanced game server to the SCC server. The SCC server stores context definitions in a repository. A call center agent, during initiation, sends a request to the SCC server to get the specified game context

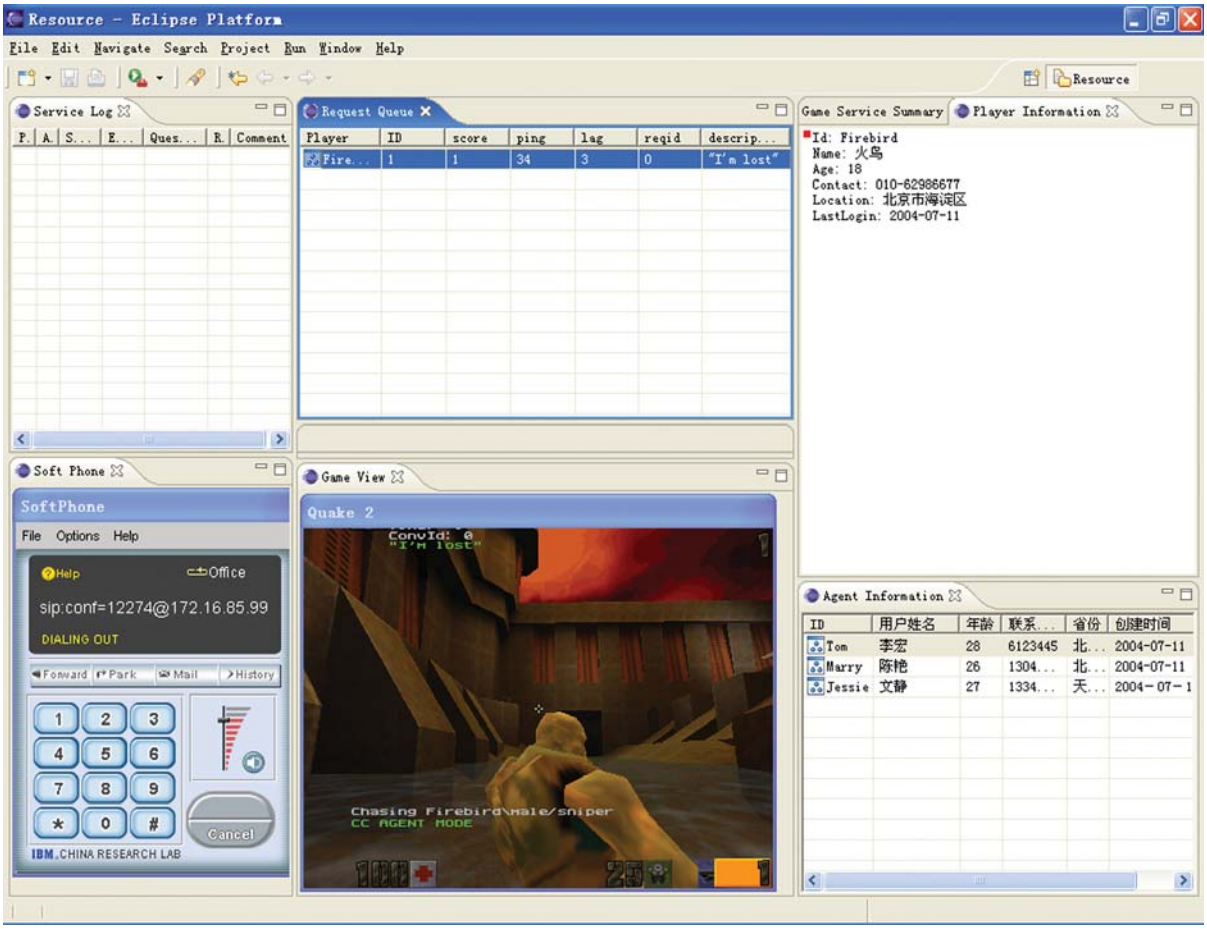

Figure 6
SCC integrated agent desktop

definition. After the context definition is acquired from the enhanced game server, each authorized SCC agent can be a game-independent agent.

"Pull mode" context acquisition

After the context definition has been downloaded, the SCC agent is ready to accept calls dispatched from the SCC server. When a game client sends a request to the call center and is the next in the queue to be served, the SCC server is responsible to query game servers and CRM servers for context messages. The SCC server then sends the combined context to an SCC agent while connecting it with the game client through a third-party call control operation in SIP.[11] This is how the SCC agent functions in a "pull" mode from the SCC server point of view.

A "push" mode can also be used to obtain context messages, in which the game client sends a request directly to the enhanced game server. In this case, the enhanced game server takes charge of initiating the process for the request that it has received. As a result, this mode requires more modifications to the game server. Because it is more difficult to modify the game server that is maintained by game operators, the pull mode is used in our implementation. An example of an XML tag used to convey SIP information is `<context_definition_ID CN04760521001/>`. The messages sent by the SCC server consist of more contextual information, which is acquired from the game server and CRM server.

System configuration

In the previous section, the logical system design and the data flow sequence of the SCC were introduced. This section focuses on the actual implementation of the SCC architecture. Based on

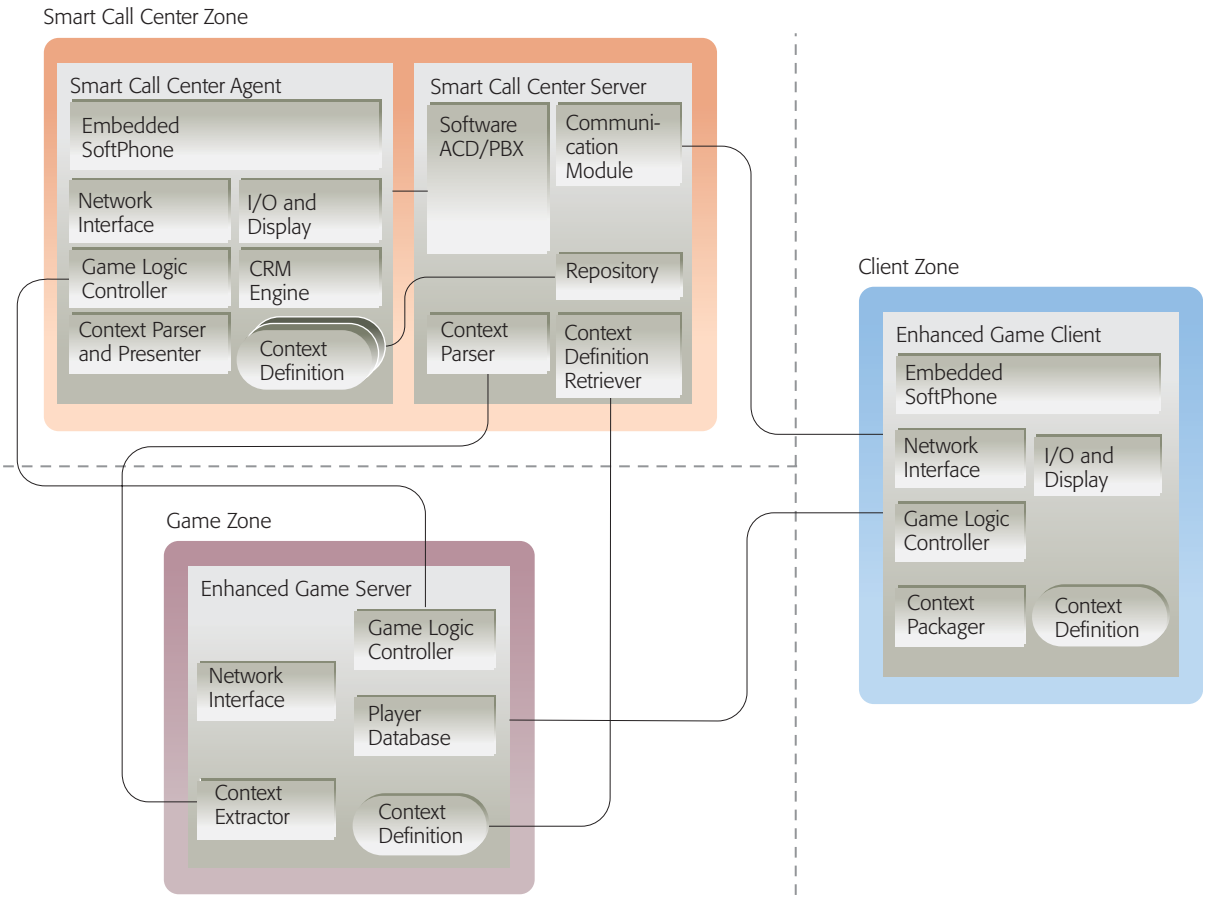

Figure 7
Component view of SCC

Quake II**, an open-source game, we developed an enhanced game client and an enhanced-game SIP-softphone-enabled server with context-analysis modules.

Enhanced game client

We extended the Quake client with two major capabilities: VoIP communication and context-aware service request capability. The original Quake client had no voice communication capability. To facilitate in-game communication and avoid the distraction of dialing a phone, we embedded a SIP softphone into the Quake II client with some extensions. The softphone is responsible for the setup, management, and termination of voice conversations with other parties. It also captures the player's voice and replays audio streams. The softphone can be launched, managed, and terminated by the extended Quake II client. After it is launched, the softphone opens a UDP (User Datagram Protocol) port to receive control messages. For example, the `HANGUP` control message causes the softphone to terminate a conversation with a peer.

The key feature of the SCC is the generation of context-aware service requests. To achieve this, there are two necessary steps taken at the client. The first step is to retrieve the player's context information, such as game ID, game server, user ID, and location, and send this information to the embedded softphone through the controlling interface. For example, a control message built with context is "`CONNECT CallCenter@9.181.115.108 Quake II 9.181.115.110 Derek hall`"; that is, a player named Derek is playing the game Quake II on the server at IP address 9.181.115.110 in the location 'hall'. The second step is to package and send the context-aware service request to the call center. After

receiving the control message, the softphone sends an `INVITE` message to the call center followed by an `INFO SIP` message with the context. An authorization packet is also sent to the SCC server to entitle it with context retrieval rights for the user Derek.

Enhanced game server

We extended the Quake II game server with open interfaces to query player information. In our prototype, we use player information only to validate a service requestor's identity. The server opens a UDP port to receive query requests from the call center server. A query request is of the format `GETPLAYER Derek/Retrieval entitlement`. After receiving a query request, the game server returns the player's identity, character role, equipment, grade, location, and other parameters to the call center server as context. In our example, the reply might be `RETURNPLAYER Derek hall`.

SCC server

The SCC server is the bridge between game player, game server, and call center agent. The prototype SCC server is an application built on the WebSphere* Application Server. After receiving the `INFO SIP` message from game client, the context parser in this server parses the XML-based message body and retrieves the player information according to the prestored context definition. In our example, the call center server then knows that a player named Derek is playing the game Quake II on the server at IP address 9.181.115.110 in the location 'hall'. Then, it sends a query request to the server at 9.181.115.110 to validate the player with the identity Derek. If the player is valid, the service request with the context is relayed to the call center agent and a context-based conversation is established between the SCC agent and the game client.

SCC agent workbench

The efficiency of the call center agent is critical for high-quality service. To improve working efficiency, we developed an integrated call-center agent workbench. This integrated workbench has a unified display of the player's information, game context view, and game client and an embedded softphone, as shown in Figure 6. It is built on Eclipse plug-in technology to leverage effective user interface management. There were two challenges in building this integrated workbench. The first involved integrating a complex game client with a workbench, a challenge because most game clients are not developed using Java** code. In our prototype, the Quake client is written in C code. To resolve this problem, the client is regarded as a native window. The handler of this window is captured and transferred to the workbench, so that it can be displayed in the workbench and obtain the user's input. The second challenge was the establishment of a common communication channel among all components in this workbench. With this channel, all components can be coordinated to facilitate the agent's service work. To achieve this target, a coordination control bus was built behind the workbench to accept messages and render them as components according to message type.

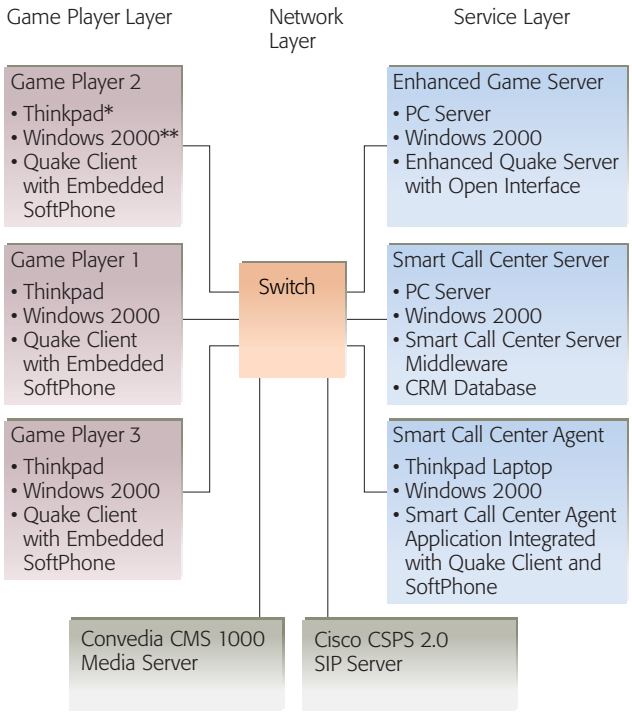

Figure 8
System configuration used in SCC analysis

SIP server and media-server configuration

We used the Cisco SIP proxy server CSPS 2.0 to act as a SIP server. It runs on an IBM x330 server with Linux** 7.2. The media server is a Convedia CMS-1000 Media Server with firmware version 4.2. The system configuration is shown in *Figure 8*.

Security considerations

Because the game server and call-center server are most likely from different companies, when an SCC server queries for game context, the game server

should authenticate whether it is a certified SCC server. Although this is not implemented in the current version of our prototype, a secure authentication mechanism is required for business implementations to avoid misuse of queries, because a player's context must be regarded as private information.

Quake II implementation summary

We based our test bed on Quake II because it is an open-source game. Another appealing feature is that it supports a "chase" mode, which enables one player to follow another player's route simultaneously. To implement the SCC solution, we modified the source code of Quake II, added two modules, and implemented a new function, `ChaseSpecific`, in the Quake server command list.

When a client calls the SCC, it first invokes the control message `CC_addRequest`, and the user-ID and service-request content are transmitted as the context. The server then displays the queue of received calls on the agent's screen using `CC_ReqsBoardMessage`. According to the queue status, when a client is ready to be served, the server notifies the related clients using `CC_accept`, opens the VoIP channel, and uses the `ChaseSpecific` command so that the view on the agent's screen is synchronized to the "chase view" of the current client.

We modified three modules of Quake II (the game base [GB], game user interface [UI], and game logic [GL]) and added two modules (SCC logic [SL] and SCC base [SB]). The total number of lines of modified or new code was 713, including 84 for the GB, 221 for the UI, 26 for the GL, 170 for the SL, and 212 for the SB. The percentage of modified code for Quake II was $(84 + 221 + 26)/200344 = 0.17$ percent, and the percentage of modified modules was $17/147 = 4.8$ percent.

The integration of the SCC with the Quake game was relatively easy. *Table 1* contains detailed code-modification information. Quake's chase mode simplified the rendering of the gaming context environment on the agent's screen, but for other online games, the development of this module may involve more implementation work.

BENEFIT ANALYSIS: QUALITY AND COSTS

In this section, we analyze the benefits of the SCC from two perspectives: quality improvement and cost reduction. For quality improvement, we use service level and waiting time as the evaluation criteria; for cost calculation, the criterion is based on the number of required agents.

Algorithm

The Erlang C Formula[12–14] was used to calculate the call-center service level with queuing. For simplicity, it assumes that customer abandonment does not occur. In the SCC solution, because calls are made by the VoIP softphone, this assumption is reasonable. Here the service level is defined as the percentage of all customers entering the system who have been helped before the maximum acceptable waiting time.

For our analysis, an Excel-based simulation tool[2] was used for the calculation of the service level. This tool takes into account the fluctuation of arrivals throughout the day by modeling the call center as a continuous Markov chain. The function used was: `callCenterValues (arrival rate, service time, numberOfAgents, numberOfLines, acceptableWaitingTime, periodLength, numberOfDesks)`, where the arrival rate (λ) is the number of arriving customers per minute. Here it is assumed that the time before an arrival is exponentially distributed. Therefore the number of customers that arrive in a certain period can be seen as a Poisson process with parameter λ. The service time (β) is the average amount of time an agent needs to help one customer. `numberOfAgents` (n) is the number of available agents in a given period. It varies over time; for example, more agents are needed during periods of heavy traffic. The number of available telephone lines at the call center is `numberOfLines`. This is the maximum number of customers that can be in the system. The call center policy for how long customers in the system may maximally wait is `acceptableWaitingTime`. The duration (in minutes) of a time segment is `periodLength`; the model allows this parameter to be chosen in order to enable the segmentation of the day in various ways. The total desks in the call center, which should be larger than the number of agents, is `numberOfDesks`.

Analysis results

The data we used for the performance analysis are based upon a Shanda call center report[5] and a 2004 iResearch survey report about online game research,[4] which shows that the Shanda call center had 80 inbound lines, 100 desks for both inbound and outbound calls, and received about 6,000

Table 1. Quake II code modification information

Module File	Total Lines of Code	Purpose
client/cl_main.c	4	Initialize VoIP module
client/cl_scrn.c	60	User interface enhancement for player and call center client
client/console.c	20	Enhanced for sending message from player to call center client
client/keys.c	5	Enhanced for sending message from player to call center client
client/menu.c	31	Add more options
client/voip.c	212	Module to communicate with VoIP softphone
game/g_callcenter.c	98	Enhanced for call center client user interface and conversation control
game/g_chase.c	10	Invoke chasing function to follow serviced player
game/g_cmds.c	180	Enhanced command control and command interface
game/g_save.c	10	Generate conversation context
game/g_spawn.c	14	Generate status information for call-center agent
game/p_client.c	35	Manage call-center agent properties
game/p_hud.c	9	Display call center information
game/p_view.c	13	Call-center agent-display enhancement
qcommon/common.c	1	Change version information
server/sv_main.c	4	Display information for client connection
win32/net_wins.c	6	Handle error situation

customer calls every day. The distribution of online game players at different time periods in a day are shown in *Table 2*, and we use the same percentage as the call center access percentage. The number of agents is an empirical one. Note that the peak hours are from 5 p.m to 1 a.m., and the number of agents at this time is doubled. The other related empirical constants are shown in *Table 3*.

Simulation 1: Performance improvement

In this simulation, we compared the performance of a normal call center with the proposed SCC solution. We assumed that the verbal description of context information takes about 1.5 minutes, which means SCC can reduce service time by this amount. Other constants were the same for the two kinds of call centers. The service level and the waiting time for the two call centers were calculated.

The results displayed in *Figure 9* show that the SCC can significantly improve the service level, especially in the peak hours between 7 p.m. and 9 p.m., where the service level is improved from 0.091 to 0.468. Waiting time is also reduced to less than 1

Table 2 Users and agents in different time periods

Period	1–3 AM	3–5 AM	5–7 AM	7–9 AM	9–11 AM	11 AM–1 PM	1–3 PM	3–5 PM	5–7 PM	7–9 PM	9–11 PM	11 PM–1 AM
User percentage (%)	5.85	6.58	2.92	4.30	7.92	5.04	7.39	7.37	10.88	18.95	15.29	7.50
Arrival rate (per minute)	2.925	3.29	1.46	2.15	3.96	2.52	3.695	3.685	5.44	9.475	7.645	3.75
Number of agents	30	30	30	30	30	30	30	30	60	60	60	60

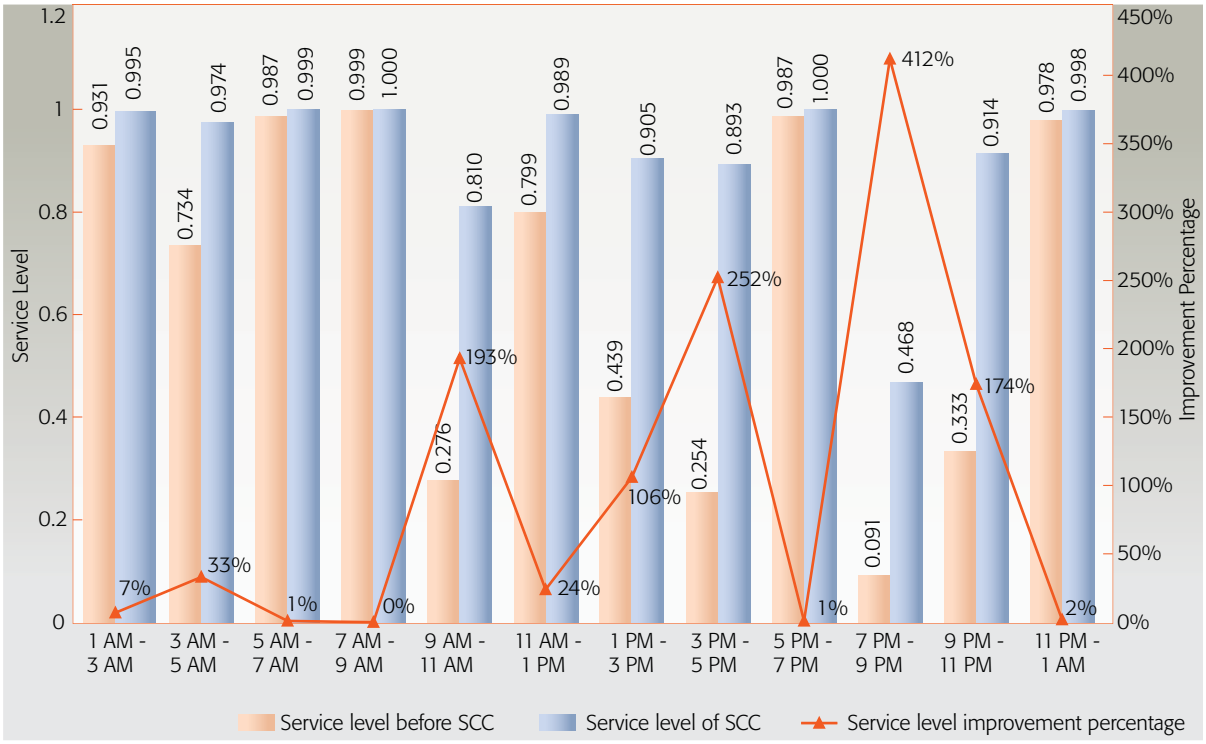

Figure 9
Service-level improvement

minute. This reduction in waiting time significantly improves the user experience when accessing call center services.

In this analysis, we assumed that the SCC can save about 1.5 minutes by automatically acquiring context. Actually, the reduction in service time obtained by the SCC may be even larger. Next, we examine the service-level improvement according to different estimates of service time reduction. We set the arrival rate at 9.5 arrivals/minute in the period from 7 a.m to 9 p.m. and let `numberOfAgents` be 60. We assume the service time for the traditional call center is 8 minutes.

Table 3 Constants for a typical call center

Constants	
Service time (minutes)	8
Number of lines	80
Acceptable waiting time (seconds)	30
Period length (minutes)	120
Number of desks	80

We allow the service time reduction obtained by the SCC to range from 0.5 minutes to 5 minutes with a step of 0.5 minutes, and test the service level at each sample point. At the service time reduction of 2.5 minutes, the service level can actually reach the general call center criterion of exceeding 0.8. This means that if SCC can improve service time by 2.5 minutes, the service level can reach the general call center criterion with the existing 60 agents.

The preceding analysis mainly verifies the quality improvement of SCC from the quantitative results. The qualitative user experience of the SCC will also be improved. Imagine that a user is in the midst of fighting a battle and wants to solicit help related to map information. An SCC agent can understand him quickly and give timely help without distracting him from the game, leading to a higher probability of winning the game and a significant increase in customer satisfaction.

Simulation 2: Cost reduction

In this simulation, we verify to what extent the SCC can reduce call center costs by evaluating the required number of agents. Two service periods are compared, 8 minutes and 6.5 minutes. For each

period, the number of agents is adjusted to make the calculated service level greater than 0.8, as recommended by the general call center criterion. The total agent working hours in a day is 928 hours for the traditional call center and 776 hours for the SCC. The cost reduction percentage is $(928 - 776)/928 \approx 16.4$ percent. If the average cost per agent is \$10/hour, then the annual cost reduction is $(928 - 776) \cdot \$10 \cdot 365 = \$554{,}800$.

Next, we calculate how many agents are required for the SCC solution in various time-saving situations during the peak hours between 7 p.m. and 9 p.m. We set the arrival rate at 9.5 arrivals/minute and let the service time of a traditional call center be 8 minutes. We then set the service time reduction obtained with the SCC from 0.5 to 5 minutes with a step of 0.5 minutes, and calculate the required number of agents at each sample point in order for the service level to exceed 0.8. The results show an approximately 22 percent reduction in the number of agents for a service time reduction of 2.5 minutes.

CONCLUSION

The context-aware SCC can combine voice-channel and game-related data during call center service and render them in an integrated environment, enabling a quick and efficient help service for online game players. It also provides a new programming model for online game engines. Our future work will focus on exploring the context acquisition model in more complex games such as MMORPGs (massively multiplayer online role-playing games). In fact, this context-aware model can be applied to any other industry that has a similar business model and uses collaboration through a centralized operation server.

ACKNOWLEDGEMENTS

The system described in this paper is a result of the Extreme Blue* project of 2004. The authors would like to thank Li Li, Wenrui Wang, Duanfeng Si, Hai Zhao, Huacheng Ke, and Jian Luo for their help in system implementation, and Mingzhe Li, Ping Qin, Jian Tang, and Susan Xu for their introduction of the IBM call center solution.

*Trademark, service mark, or registered trademark of International Business Machines, Inc.

**Trademark, service mark, or registered trademark of Sun Microsystems, Inc., Linus Torvalds, Id Software, Inc., or Microsoft Corporation in the United States, other countries, or both.

CITED REFERENCES

1. *Transformation in the Contact Center Environment*, IBM Business Consulting Services (2004), http://www-1.ibm.com/services/us/bcs/pdf/g510-3642-contact-center-trends.pdf.
2. IBM Call Center Continuity Services, http://www-1.ibm.com/services/us/index.wss/offering/bcrs/a1000262.
3. Cisco IP Contact Center Enterprise Edition, Solution Reference Network Design (June 2005), http://www.cisco.com/application/pdf/en/us/guest/netsol/ns268/c649/ccmigration_09186a00803fecb5.pdf.
4. *2004 China Online Game Research Report* (in Chinese), iResearch, http://www.iresearch.com.cn/online_game/detail_free.asp?id=11653.
5. Shanda Entertainment (2005), http://www.snda.com/.
6. "Customer Service: The Key to the Success of Online Games," (in Chinese), http://games.sina.com.cn/newgames/2003/09/09246058.shtml.
7. Q. Ren and X. Yang, "Analysis of the Development of the Chinese Online Game Industry," Online publication (2004), http://www.mindtrek.org/liitetiedostot/materiaalit_editori/142.pdf.
8. H. Bernett and M. L. Jaramillo, "Assessing Web-Enabled Call Center Technologies," *IT Professional* **3**, No. 3, pp. 24–30 (May/June 2001).
9. G. Koole, "Performance Analysis and Optimization in Customer Contact Centers," *Proceedings of the First International Conference on the Quantitative Evaluation of Systems (QEST'04)*, IEEE Computer Society Press, Los Alamitos, CA (2004), pp. 2–5, http://www.math.vu.nl/~koole/articles/qest04/.
10. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, *SIP: Session Initiation Protocol, RFC 3261*, Internet Engineering Task Force (2002), http://www.ietf.org/rfc/rfc3261.txt?number=3261.
11. J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo, *Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP), RFC 3725*, Internet Engineering Task Force (2004), http://www.ietf.org/rfc/rfc3725.txt?number=3725.
12. G. Koolel and A. Mandelbaum, *Quantitative Management of Call Centers*, Vrije Universiteit, Amsterdam, Department of Mathematics, http://www.cs.vu.nl/obp/callcenters.
13. Telecom Traffic Online, Westbay Engineers, Ltd., http://www.erlang.com.
14. *CallVusion DSS, OBP Decision Support Systems*, Vrije Universiteit, Amsterdam, Department of Mathematics, http://www.cs.vu.nl/~obpdss/obpdss.php?action=info&subject=callVusion.

Accepted for publication October 11, 2005.
Published online January 20, 2006.

Lin Luo
IBM Research Division, IBM China Laboratory, Building 19, Zhongguancun Software Park, 8 Dongbeiwang West Road, Haidian District, Beijing 100094, People's Republic of China (luolin@cn.ibm.com). Dr. Luo is a research staff member in the Digital Media department at the China Research Laboratory. She received a Ph.D. degree in electrical engineering from the University of Science and Technology of China in 2003. She subsequently joined IBM at the China Research Lab, where she has worked on digital media processing and

transmission. She was one of the technical mentors of the 2004 Extreme Blue program on the topic of VoIP enabling in online games.

Jun Liu
IBM Research Division, IBM China Laboratory, Building 19, Zhongguancun Software Park, 8 Dongbeiwang West Road, Haidian District, Beijing 100094, People's Republic of China (liujunlj@cn.ibm.com). Dr. Liu is a research staff member in the On Demand Technology department at the China Research Laboratory. He received a Ph.D. degree in electrical engineering from the Beijing University of Posts and Telecommunications in 2003. He subsequently joined IBM at the China Research Lab, where he has worked on converged network and services technology. He was one of the technical mentors of the 2004 Extreme Blue program on the topic of VoIP enabling in online games.

Ling Shao
IBM Research Division, IBM China Laboratory, Building 19, Zhongguancun Software Park, 8 Dongbeiwang West Road, Haidian District, Beijing 100094, People's Republic of China (shaol@cn.ibm.com). Mr. Shao is the manager of the Digital Media department at IBM's China Research Laboratory, which he joined in 2000. His major research interests are multimedia processing, multimedia systems and solutions, and information security. He has filed more than 10 patents and published several conference papers in these areas. He received a Master's degree in electrical engineering from Fudan University in China, in 2000.

Wei Lu
IBM Research Division, IBM China Laboratory, Building 19, Zhongguancun Software Park, 8 Dongbeiwang West Road, Haidian District, Beijing 100094, People's Republic of China (luw@cn.ibm.com). Dr. Lu is the manager of the On Demand Technology department at the China Research Laboratory. She is an advisory research member. She obtained a Ph.D degree in computer software at Tsinghua University, Beijing, People's Republic of China and has worked at IBM for four years. Dr. Lu's research interests include network and service infrastructure, Internet multimedia and telephony, and telecom service delivery platforms. She is the leader of the SIP Research Institute.

Meng Ye
IBM Research Division, IBM China Laboratory, Building 19, Zhongguancun Software Park, 8 Dongbeiwang West Road, Haidian District, Beijing 100094, People's Republic of China (yemeng@cn.ibm.com). Dr. Ye is a research staff member in the High Performance Computing group at IBM's China Research Laboratory. He received a B.S. degree in computer software from Sichuan University in Chengdu, China in 1991, an M.S. degree in computer science from Southwest Jiaotong University in Chengdu, China in 1994, and an M.Phil. degree in information systems from the City University of Hong Kong in 1999. He worked for the Institute of Computing Technology of the China Academy of Science before joining IBM in 1999. ■